CSE 160 Section 10

Classes

1. Given the following class Shop :

```
class Shop:
    '''
    A class which represents the contents of a shop.
    A Shop stores the shops name as well as the inventory
    '''
    def __init__(self, name):
        '''
        Creates a new shop with a given name, but an empty inventory
        name must be a string
        '''
        # name is a string representing the shop
        self.name = name
        # inventory is a dictionary mapping each item to the amount
        # in the shop
        self.inventory = {}

    def print_inventory(self):
        '''
        Prints this shops inventory, where each item is followed
        by the amount of that item in the shop
        '''
        for item, amount in self.inventory.items():
            print(item, amount)

    def add_item(self, item):
        '''
        Adds an item to the store's inventory
        '''
        if item not in self.inventory:
            self.inventory[item] = 0
        self.inventory[item] += 1
```

1.a. What would the following client code print?

```
flower_shop = Shop(“Swansons”)
flower_shop.add_item(“Daisy”)
flower_shop.print_inventory()
```

Daisy 1

1.b. What code would you write to make a new shop called Arcane Comics that has 5 comics in it, and then print out it's inventory?

```
comic_shop = Shop("Arcane Comics")
for i in range(5):
     comic_shop.add_item("Comic")
comic_shop.print_inventory()
```

1.c. Adding one item at a time is extremely tedious. Write a function for the class Shop called `add_items` that given an item and an amount, adds that amount to the store's inventory

```
def add_items(self, item, amount):
     if item not in self.inventory:
         self.inventory[item] = 0
     self.inventory[item] += amount
```

CSV DictReader and Matplotlib

2. Say we have the csv file “crispra_data.csv” that came from a lab that does research on CRISPR, a gene editing/regulation tool. In this data, an experiment was performed where scientists were testing some conditions and measuring the concentration of green fluorescent protein (GFP) at various time steps.

It might look something like this:

hours	**GFP Rep 1**	**GFP Rep 2**	**GFP Rep 3**
0	13105.08	12817.37	14477.5
1	8287.21	8761.215	9498.602
...	...	...	...
5.883333	17117.91	17779.74	17797.61

(You can find this file and a skeleton starter file here: https://tinyurl.com/cse160ad10)

2.a. Create a function called `extract_column` that will take in a filename and a column name, and return a list with all the data from that column as floats. **Hint:** Use CSV DictReader

```
def extract_column(filename, column_name):
      csv_file = open(filename)
      csv_dict = csv.DictReader(csv_file)

      lst = []
      for row in csv_dict:
                col_val = row[column_name]
          lst.append(float(col_val))

      csv_file.close()

      return lst
```

2.b. For each repetition of the experiment (GFP Rep 1, GFP Rep 2, and GFP Rep 3) plot it on a graph. The x axis should be the hours, the y axis the concentration of GFP.
Each repetition should be a different line on the graph. Don't forget to give each line a label. You should also give a title to the plot. You should save the plot as well.

```
# Get the data
hours = extract_column("crispra_data.csv", "hours")
gfp1 = extract_column("crispra_data.csv", "GFP Rep 1")
gfp2 = extract_column("crispra_data.csv", "GFP Rep 2")
gfp3 = extract_column("crispra_data.csv", "GFP Rep 3")

# plot
plt.plot(hours, gfp1, label="Rep 1")
plt.plot(hours, gfp2, label="Rep 2")
plt.plot(hours, gfp3, label="Rep 3")

# give it a title + labels
```

```
plt.title("GFP Concentration Over Time")
plt.ylabel('GFP Concentration')
plt.xlabel('Time (hours)')

# show the legend
plt.legend(loc='best')

# save the graph
plt.savefig('GFP Graph')

# clear the current figure -- say if we wanted to make new
# graphs later in the program
plt.clf()
```